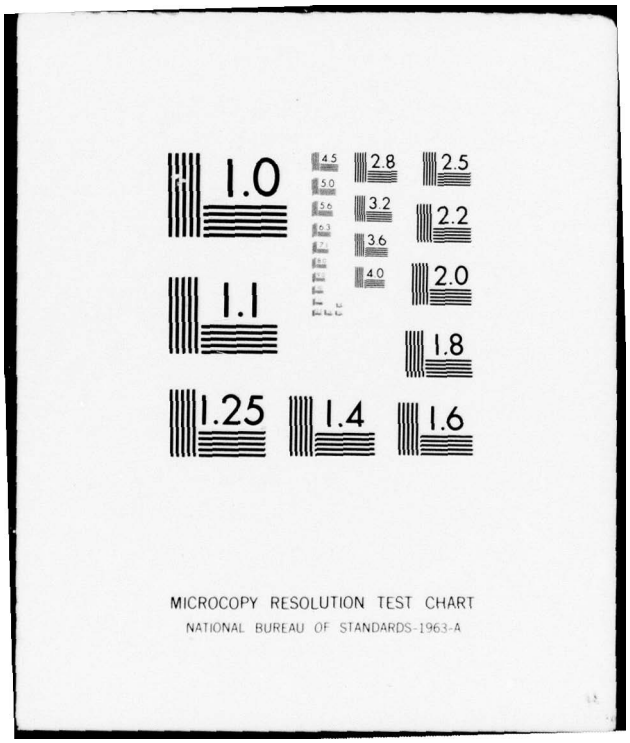
AD-A040 329 HONEYWELL INFORMATION SYSTEMS INC MCLEAN VA FEDERAL --ETC F/G 9/2
ENGINEERING INVESTIGATIONS IN SUPPORT OF MULTICS SECURITY KERNE--ETC(U)
OCT 76 N ADLEMAN F19628-74-C-0193

UNCLASSIFIED ESD-TR-77-17 NL

| OF |
AD
A040329

END

DATE
FILMED
7-77


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
5.6
6.3
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

ESD-TR-77-17

ENGINEERING INVESTIGATIONS IN SUPPORT OF MULTICS SECURITY KERNEL SOFTWARE DEVELOPMENT

Honeywell Information Systems, Incorporated
Federal Systems Operations
7900 Westpark Drive
McLean, VA 22101

October 1976

Approved for Public Release;
Distribution Unlimited.

Prepared for

DEPUTY FOR COMMAND AND MANAGEMENT SYSTEMS
ELECTRONIC SYSTEMS DIVISION
HANSCOM AIR FORCE BASE, MA 01731

LEGAL NOTICE

When U. S. Government drawings, specifications or other data are used for any purpose other than a definitely related government procurement operation, the government thereby incurs no responsibility nor any obligation whatsoever; and the fact that the government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data is not to be regarded by implication or otherwise as in any manner licensing the holder or any other person or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

OTHER NOTICES

Do not return this copy. Retain or destroy.

This technical report has been reviewed and is approved for publication.

William R Price
WILLIAM R. PRICE, Captain, USAF
Techniques Engineering Division

Roger R. Schell
ROGER R. SCHELL, Lt Col, USAF
ADP System Security Program Manager

FOR THE COMMANDER

Frank J. Emma
FRANK J. EMMA, Colonel, USAF
Director, Computer Systems Engineering
Deputy for Command & Management Systems

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: ESD-TR-77-17
2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle): ENGINEERING INVESTIGATIONS IN SUPPORT OF MULTICS SECURITY KERNEL SOFTWARE DEVELOPMENT,
5. TYPE OF REPORT & PERIOD COVERED
6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s): N. Adleman
8. CONTRACT OR GRANT NUMBER(s): F19628-74-C-0193

9. PERFORMING ORGANIZATION NAME AND ADDRESS
Honeywell Information Systems, Incorporated
Federal Systems Operations
7900 Westpark Drive, McLean, VA 22101
10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: CDRL Item A005

11. CONTROLLING OFFICE NAME AND ADDRESS
Deputy for Command and Management Systems
Electronic Systems Division
Hanscom AFB, MA 01731
12. REPORT DATE: October 1976
13. NUMBER OF PAGES: 33 (12 30p.)

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)
15. SECURITY CLASS. (of this report): UNCLASSIFIED
15a. DECLASSIFICATION/DOWNGRADING SCHEDULE: N/A

16. DISTRIBUTION STATEMENT (of this Report)

Approved for Public Release; Distribution Unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

MULTICS
Computer Security
Multilevel Security
Security Kernel

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report provides the status of certain engineering efforts to support the development of a secure general purpose computer.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

P R O J E C T G U A R D I A N

Engineering Investigations in Support of Multics Security Kernel Software Development

19 October 1976

prepared for

Electronic Systems Division
Air Force Systems Command
L. G. Hanscom Air Force Base
Bedford, Massachusetts 01731

Contract No. F19628-74-C-0193

CDRL Sequence No. A005

Honeywell Information Systems Inc.
Federal Systems Operations
7900 Westpark Drive
McLean, Virginia 22101

PREFACE

This report provides the status of certain engineering efforts to support the development of a secure general purpose computer. The report states that these efforts are part of a multi-year development to produce a prototype secure Multics. In August 1976, while this report was in preparation, the Air Force terminated the development of the secure Multics because of funding limitations.

Section I

Introduction to the Computer Security Program

The problem of security in computer systems has been under study for several years. The Air Force has sponsored studies and development projects aimed at improving understanding of security in computer systems, developing a sound tneoretical basis for further work, and demonstrating accomplishments in the field. Many of these projects have been associated with the Multics system.

The overall goal of these efforts has been to develop a certifiably secure computer system for general use by the military to meet their operational requirements. This program, known as Project Guardian, will take the Multics system from its present form to a prototype Multics system which can be used to demonstrate the feasibility of software certification. A necessary step in this transformation is the development of a restructured Multics which operates on a software structure known as a security kernel. Other activities are the development and integration of a Secure Front-End Processor (SFEP), and the identification of the technology and tools which will allow the actual certification of the Multics and SFEP software kernels.

Background

The military faces an increasing need for operational computer systems capable of processing several levels of classified information at the same time. Present systems (except Multics) are unable to support secure multilevel processing due to fundamental weaknesses in their basic design since security was not a concern when they were developed. The weakness is that current hardware/software systems are unable to adequately protect the information that they process.

With the exception of the two-level Multics system developed for, and in use at, the Air Force Data Services Center, the military currently meets the need for processing several levels of information by one of two methods. Either all security levels are processed together at the level of the highest classification present, or each level is processed by itself. Both methods have been less than satisfactory. The problem with processing all levels together is that all users and all equipment, including terminals and communications facilities, must be cleared to the highest classification that the system can ever process. The problem with separate processing is that a separate computer system or a separate period of time is required for each level handled. Either method is costly and inefficient. Neither method allows simultaneous handling of information at several levels for users of several levels of clearance.

Multics is the most advanced general computing system as far as security is concerned. Security was one of the initial design goals of the Multics system designers and has been a major concern of the designers and developers throughout the history of the system. Even with this concern for security, the present Multics system cannot be certified secure. Multics, however, does present the best available base upon which to build a certifiably secure multilevel computer utility.

Secure communications has also presented operational problems to the military. A secure on-line system requires a secure communications network. While the techniques of securing communications lines and terminals have been well developed, a certifiably secure communications processor is still undeveloped. A secure multilevel system must have a compatible and secure front-end communications processor to be able to properly handle multiple levels of classified information.

Both economic and operational considerations make development of a certifiably secure multilevel system desirable. Recent advances in computer technology indicate that it should be possible to produce a system that can process an arbitrary mix of classified and unclassified information simultaneously on a single computer system. The system should serve both cleared and uncleared users and should rely on the computer system's internal hardware/software controls to enforce security and need-to-know requirements. Of primary importance is that the system be certifiably secure. That is, it must be possible to prove that the system is complete and without flaw in any of its security-related aspects.

The Air Force has been working on the problem of providing a certifiably secure multilevel system. In 1970, the Air Force Data Services Center (AFDSC) requested the Electronic Systems Division (ESD) to support development of an open multilevel system for the AFDSC Honeywell 635 systems. The resulting studies pointed out the severity of the problem and led to the formation of a computer security technology planning study panel. The panel's report (1) described the fundamental problems and delineated a program to develop the desired system. The panel recommended that the technical approach to the problem be "to start with a statement of an ideal system, a model, and to refine and move the statement through various levels of design into the mecnanism that implement the model system".

The basic component of the ideal system was also identified by this panel. This component is known as the Reference Monitor, an abstract mecnanism that controls access of subjects (active system elements) to objects (units of information) within the computer system and enforces the rules of the military security system on such access. Three requirements were recognized for a Reference Monitor:

1. Complete Mediation - the mechanism must mediate every access of a subject to an object.

2. Isolation - the mechanism and its data bases must be protected from unauthorized alteration.

3. Verifiability - the mechanism must be small, simple, and understandable so that it can be completely tested and verified (certified) to perform its functions correctly.

The mechanism that implements the Reference Monitor in a particular computer system has been termed the security kernel. Much subsequent work has been devoted to identifying the characteristics of a security kernel and to exploring the technology involved in producing a security kernel for some computer system. (This report summarizes the more significant research tasks which are evolving the present Multics supervisor into a security kernel for Multics).

ESD initiated development of formal mathematical models of the ideal Reference Monitor in 1972. This work (2,3) resulted in a model of a secure computer system as a finite-state mechanism that makes explicit transitions from one secure state to another. The rules of the model formally define the conditions under which a transition from state to state can occur. The rules have been proven to allow only transitions that preserve the security of information in the system. The model specifies requirements for the operation of a security kernel. These requirements were taken directly from the Defense Department regulations on handling sensitive information (DoD Directive 5200.1-R). With the availability of the model, the problem of validation is now reduced to providing complete assurance that a particular security kernel behaves exactly as the model requires.

Work on the technology of certification progressed in parallel with the work on the model. In 1973, W. Price (4) identified a methodology for verification of a kernel. More detailed developments of this validation methodology have been reported by MITRE (5,6). Another approach has been explored which may be more suitable to large software modules (7).

Other activities have been devoted to the problem of building a security kernel for a practical system (8,9). This work has demonstrated the soundness of the basic concepts and also pointed out some of the problems that lie in the way of realizing a security kernel on a large system. This work has been the basis for development of a secure communications processor, a detailed effort which is an integral part of the computer security program.

A major project in the development process is the development of a security kernel for a large resource sharing system. The

system chosen for this effort is Multics. There are two reasons that this choice was made. First, the hardware base of the Multics system, the Honeywell 68/80 computer, has been identified as best suited of all off-the-shelf large computer systems for the support of a security kernel (10). Second, the Multics system architecture was conceived and developed with security requirements specifically in mind.

One project, now completed, involved the design and production of a Multics system capable of supporting a two-level (Secret and Top Secret) environment for the Air Force Data Services Center (11,12). This system implements security controls based on the military access rules, but it does not completely handle the threat of a hostile penetration. From these efforts, additional insight was gained in the problems of designing and developing a security kernel for Multics.

Technical Approach of Project Guardian

Design of a security kernel for Multics is currently underway as a joint effort between Honeywell, ESD, the MITRE Corporation, the Massachusetts Institute of Technology (13), and Stanford Research Institute. This design effort has led to a more complete understanding of the general problem and has provided the foundation for the overall development plan in "Multics Security Integration Requirements, 1976 - 1980" (14). Work is progressing on formal specifications for the Multics security kernel (15) (16) and on simplification and reorganization of the Multics operating system. The technical approach is to refine the current Multics system and reimplement critical portions of the system to produce a certifiable kernel which will interface with a certifiable front-end communications processor with its own security kernel. The result will be a prototype Multics system which may meet the goal of Air Force certification.

The entire project can be described in terms of three coordinated development activities: development of hardware for a prototype Secure Front-End Processor (SFEP) and Interface Units; development of a security kernel for the Secure Front-End Processor; and development of a certifiable prototype secure Multics including the Secure Front-End Processor.

These development activities encompass a five year effort during which major technical milestones will occur as follows:

1. Preliminary Multics Demonstration

   Specification, design, implementation, and demonstration of the first kernel-based Multics system. This system is coded in PL/I and utilizes a DATANET-6600 communications processor.

2. SFEP Development Unit

   Delivery of SFEP system hardware to the hardware system test software development facility at Honeywell's Aerospace Division.

3. SFEP Prototype System

   Specification, design, and assembly language implementation of a prototype SFEP software system.

4. TEMPEST Tested SFEP Hardware

   Availability of prototype SFEP hardware which conforms to TEMPEST requirements.

5. System Programming Language

   Availability of operational compilers which facilitate program correctness proofs to support the reimplementation of the Multics and SFEP kernels for the operational prototype Multics demonstration.

6. Intermediate Multics Demonstration

   Demonstration of an integrated, PL/I-coded, kernel-based Multics system with SFEP capabilities.

7. Tools for Correctness Proofs

   Selection of correctness proving tools to facilitate certification of the kernels in the operational Multics prototype.

8. Begin Correctness Proofs

   Begin feasibility demonstration for proof of correspondence between the specification and kernel software implementations.

9. Operational Prototype Multics Demonstration (Recoding and System Integration)

   Reimplementation of the security kernels in the selected system programming languages and implementation of performance enhancements.

10. Correspondence Technology and Specification Proofs

    Completion of the correspondence proofs between the model and the kernel specifications for the prototype using the available correspondence proof tools.

11. Establish Secure Development Site

    Availability of a secure computer facility which provides a Top Secret controlled environment in which software development, system tests, and certification can be undertaken.

12. Operational Test and Evaluation Site

    Availability of an operational Multics site which will provide the environment for the Test and Evaluation of the secure Multics system under operational service conditions.

To develop a security kernel for Multics (Task I, Preliminary Multics Demonstration above) re-engineering of the current (standard) Multics is required. One of the first re-engineering steps is the investigation into the identification of which selected components of the Multics supervisor can be reduced in both size and complexity to achieve the minimum kernel. (13) The plan is to evolve Multics into a prototype system which supports the same essential features currently available. To achieve this restrucuring, intensive research efforts into variuos elements of Multics have been experimentally undertaken. The status of these research tasks is described in Section II of this paper.

The resulting system will contain a small and simple central core of software which comprises a security kernel that can be offered for certification. The kernel will enforce access constraints that combine nondiscretionary controls reflecting the information release policies of the military security system and the discretionary controls on information sharing that continue to be part of the Multics design (17).

Section II

Task Status

The following research tasks are in varying stages of development at MIT. Some tasks have been completed; others are still being investigated. Honeywell reviews these tasks for their continued appropriateness to the evolving kernel-based Multics. In this review, the following technical criteria are used:

1. effect on system security
2. performance impact
3. compatibility at all interfaces
4. extent of required re-engineering

Some tasks have yielded definite results. Others have produced additional insights for further design investigations whose immediate results cannot be verified until the entire kernel is implemented for Guardian. As the development of the kernel-based Multics progresses, the technical results of these research tasks, whether the task is considered complete or still under investigation, will be xxxx into the kernel design as appropriate.

Completed Tasks

The tasks listed below have been completed by the Computer Systems Research Division of Project MAC.

1. Removal of the Linker from Ring 0

This task is an important first step in pruning unnecessary programs from the portion of the system which must be certified. Several other components of the system, in particular the management of reference names, can be removed from the kernel after the linker has been removed. The project also produced a MIT Technical Report and a published paper [18,19].

The initial version of the user ring linker showed 7 to 10 percent slower performance than the standard system. This was improved to show performance equal to the standard system. This task showed that the removal of the linker from ring 0 can be done and in an efficient manner. The concepts of this task will be used and will provide a simplification of ring 0. Honeywell will utilize the technical results of this task in the kernel-based Multics for Project Guardian.

2. Removal of Name Space Management from Ring 0

This task removed from the supervisor the facilities for managing the association between reference names and segments in the address space of a process. The association between names and segment numbers is now maintained in the user ring rather than in ring 0, leaving in the supervisor only the association between segment numbers and unique ID.

Removing the reference name management mechanism from the supervisor required that a data base central to the management of the address space of a process - the known segment table or KST - be split into a private and a common part, and that the supervisor learn to lie convincingly on occasion about the existence of certain file system directories. The result of the removal is a reduction by a factor of five in the size of the protected code needed to manage the address space of a process. Another result is a new simpler interface to the file system portion of the supervisor. Instead of identifying a directory with a character string tree name, a segment number is now used. The algorithm for following a tree name through the directory hierarchy to locate the named element is thus removed from the supervisor. Performance tests indicate that the new design outperforms the old.

The task has been described in a Project MAC Technical Report [20]. The technical results have shown the approach is both viable and efficient. The concepts developed from this investigation will be incorporated into the kernel-based Multics for Project Guardian.

3. Fast Processes in Ring 0

One approach to understanding and simplifying the structure of ring 0 is to separate portions of the supervisor into separate processes. It is necessary that there be available a class of process which is very inexpensive to run so that the separation could be accomplished. This task has involved the design and implementation of such fast processes. A special kind of process has been developed which runs only in ring 0, which has limited capability, and is very efficient to execute.

The fast process is one which makes very restricted demands on its environment. The process has a legitimate stack, can abandon the processor by means of the wait and notify mechanism, and can take page faults, but is restricted from taking segment faults or adding segments to its address space. The wired storage required for these processes has been reduced by two strategies. First, there is one descriptor segment per processor, used by any of these fast processes when it runs. Second, the Process Descriptor Segment (PDS) has been split into two components, only one of which is needed for these processes.

Approximately one-fourth of a page of storage is required for these processes with the rest of the page available for the stack. Thus if a fast process requires less than three-quarters of a page of stack, there is only one page required in core when that process runs.

Fast processes were tested in use by rewriting the interrupt side of the terminal device interface module so that it ran as one of these processes rather than directly as a result of an interrupt. Nine pages were able to be unwired as a result of this change. An initial version of this terminal manager process has run for an extended period in a trial xxxxx of a development system. This test implementation will not be used in the standard system since a new communications package has replaced the particular terminal manager used. This concept has been proven to support selected ring 0 system functions. The actual functions to be supported for Guardian are highly dependent upon the actual functions and design structure to be selected as resident in the security kernel. At the present time, Honeywell is evaluating the top-level functions to be included in the kernel at this time. During the development of the kernel, Honeywell will use the technical results of this task as an engineering tool to support selected functions when appropriate for the kernel.

For further experimentation, Honeywell has implemented a version of this technical work which supports the syserr-logging mechanism. As further results become available, applications of this task to additional ring 0 functions will be identified.

4. High-Level Description of System Functional Capability

As part of any attempt to certify a system, it is necessary to have some description of the intended functionality of the system itself to serve as a standard against which to certify. Various notational schemes have been tried for describing the aspects of various parts of the system. A representation of system data bases and related algorithms in the Vienna Definition Language was performed using the known segment table as a case study. A similar description of directory control, using English as the descriptive language, was also performed. Finally, the language was devised for describing programs with complexly structured data bases, which attempts to avoid implications concerning the implementation of the data base structure [21]. For the top-level specification of the Multics security kernel, Honeywell has selected the SRI-developed SPECIAL language to express the kernel concepts. The SPECIAL language provides an easier mode of expression and lends itself more readily to formal certification techniques.

5. Study of the Removal of User I/O from Ring 0

A strategy has been developed for handling user-initiated peripheral I/O which operates almost completely in the user ring. The only function which is required within the kernel is the management of multiplexed devices. The scheme uses as the buffering strategy for I/O the virtual memory management algorithm of the system. The scheme effectively removes I/O from the kernel; however, it requires an I/O controller with capabilities slightly greater than the one currently available on Multics. Thus this particular scheme will not be implemented in the near future. As the development of the kernel-based Multics progresses, the results of this technical task will be incorportated into the design.

Continuing Tasks

The tasks listed below remain active and are being pursued by the Computer Systems Research Division of Project MAC.

1. Restructuring of Page Control

Research is continuing on various ways to reorganize page control. Using the language devised under the task "High Level Description of System Functionality", a version of page control was constructed which handled read-write sequences in a separate process. This approach was then further refined to produce a version of page control which uses separate asynchronous processes to execute all of the page control functions except the act of fetching the missing page. It is felt that by isolating functions in separate processes, and constraining them by restricting the interprocess communication paths, that it will be easier to understand and certify the overall algorithm. One of the other benefits of structuring page control in this way is that it should be possible for several processors to service page faults simultaneously, without interfering with each other.

The goal of this task is to utilize several asynchronous parallel processes to perform the functions of page control. Separate processes are used to remove pages from memory and from the paging device so that a free storage pool will always exist to be used for the servicing of page faults. The processes used are examples of the fast processes developed under the above completed task "Fast Processes in Ring 0". Use of parallel processes provides simplification of the algorithm, since it eliminates some artificial interactions that occur if the functions are performed as part of the same process and which constrain the functions to run in a particular synchronized order.

Design of a demonstration version of page control has been finished and coding is well under way. The initial, experimental implementation of this task has shown this technical approach to result in an insufficient version of page control. If this design were implemented in the kernel-based Multics, system performance would be intolerable. The research data from this task will continue to be evaluated as the design for the security kernel progresses. The engineering approach of this task will be used if the implementation of page control for the kernel is significantly different from the current ring 0 such that a re-implemented and improved page control would offer a more attractive alternative.

2. Restructuring of Traffic Control

Techniques are currently being explored to restructure and simplify the traffic controller in order to speed up the act of

switching from one process to another and to simplify the mechanisms involved. The intent is to split the traffic controller into two parts, separating out the actual act of switching from one process to another from the more complex act of deciding which process is eligible to run. The division into policy and mechanism should make the algorithm easier to understand.

A design is being pursued which will retructure the traffic controller into two levels. The lower level multiplexes the real processors of the system among a fixed number of so called virtual processors. By fixing in advance the number of such virtual processes, this low level processor multiplexor need make no use of the systems virtual memory facilities. Thus there is a strict isolation and ordering between the multiplexes and the virtual memory. A higher level scheduler multiplexes some of the virtual processors among all of the currently operating real Multics processors. This higher level scheduler can use all of the facilities of the Multics virtual memory, since they are implemented at a lower level. It is expected that this restructuring will clarify the relationship between traffic control and page control and also aid in separating the idea of interprocess signaling from the idea of traffic control. In this task, no ring 0 data base (such as the current message table) will be needed for messages between processes. Messages between processes will be sent using segments that are protected using the standard system access control mechanisms. This appears to be a great simplification over the current mechanism.

Progress is being made in four areas toward the development of a demonstration version. First, the low level scheduler which implements virtual processors using real processors is being implemented. Second, the high level scheduler, which will multiplex these virtual processors among real Multics processes is being designed. Third, all portions of the system, other than traffic control, which must be modified or redesigned in order to run the new traffic controller have been identified. Included are modifications to interrupt and fault handling, changes to page fault handling, and various other small system changes. Recoding is in process. Fourth, a proposal has been prepared to eliminate from the system the traffic control data base known as the Interprocess Transmission Table (ITT). This removal would also simplify the traffic controller. This task is a significant first step to subdivide traffic control into various levels of abstraction (i.e., separate policy from mechanism). To date, incomplete data on the impact of this approach on overall system efficienty is not available. Honeywell will review this work during the kernel design to determine if this task is an appropriate engineering approach for the security kernel.

3. Restructuring of the Answering Service

The answering service is made up of those algorithms which authenticate the user, create processes, and manage teletype lines. This is a large interconnected set of functions, all of which are security sensitive. A rearrangement of the answering service has been proposed which will provide an isolation of those particular components that are in fact crucial to assure secure operation of the system. There is a similarity between between the creation of a new process and the entering of a new protection domain. This allows access control lists to be used to regulate the creation of processes on the behalf of any particular user. In general, this scheme avoids the need for certified software by providing the means to assure the user that a process created with the user's identification will start executing only in certain specified programs that the user provides. These programs are provided with tools which allow them to determine that the process has been brought into execution under appropriate circumstances.

This task has finished with the design phase and is currently undergoing implementation. Complete technical data on this task is not yet available. Honeywell will review this work during the kernel design to be certain that there is no negative impact on the user interface and system performance. This review will occur when the technical design and experimental data becomes available.

4. Multics System Initialization

If one is to certify that a system works correctly, one must begin by verifying the "initial state" of that system. For this reason it is very important to understand how the Multics system initializes itself. The current initialization of Multics is relatively unstructured and is apparently not amenable to verification or certification. This task will restructure system initialization to reduce the amount of code in the initialization phase and simplify the task of verifying the remainder.

The approach is to recognize that much of what is now considered initialization ought rather to be considered as reconfiguration. Initialization is then decomposed into two phases. The first phase involves getting a minimal Multics up and running. The second phase is a series of reconfigurations based on input describing the actual configuration in use. The advantage of this strategy is that all of the reconfigurations run in a complete, operational Multics environment, which is much easier to understand than a partial and ever-changing environment as presented in the various stages of the current initialization approach. Also, since the minimal Multics is independent of the actual configuration (given the minimum hardware required), it can be largely generated as the system tape is created. Algorithms which run at the time that the system tape is created are easier to verify since they too run in a fully operational

Multics environment.

Detailed design of an experimental and reduced portion of a restructured system has been done. Coding and design verification is underway. Initialization of the security kernel for the Guardian Multics is xxx addressed as an integral element of the total kernel design progress. This research work has provided the first investigation of Multics initialization from a security viewpoint. This work will be further reviewed a additional data becomes available and then overall kernel design progresses.

5. Multitasking in the User Ring

This task will provide an environment that will allow the user to write various programs as if they were executing in separate cooperating processes. The execution of these processes is supported by multiplexing the one single process of the user. Thus, this is described as multitasking rather than as multiprocessing. The creation of this program environment in the user ring does not directly contribute to simplification of the kernel, but the existence of the facility will allow other modifications that will simplify the kernel. Among these is the design of a simple (but effective) QUIT handling mechanism to replace the current complicated one, the multiplexing of network server processes, and the restructuring of the answering service.

One experiment involved implementation of a version of user ring Interprocess Communication (IPC) as part of an experimental command processor. In this version, the command processor runs each command in a separate task. When a QUIT is signaled in the user's process, the result is that only one task need be suspended and control returned to the listener. Any new task (command) started will run on a different stack, thus signal handlers for various tasks never become confused. It is possible to restart any task in any order with any number of other tasks suspended.

The multitasking environment in the user ring shows a means of obtaining some second-order effects from simplifying the kernel (e.g., simplification of QUIT handling). These second-order effects of this task may be nullified (e.g., loss of compatibility at the user interface) as the kernel design progresses.

6. A Methodology for Designing Computer Systems

This task has been concerned with developing a general methodology for designing certifiably secure systems. The goal is to make the systems easier to certify as correct. A method has been formulated and is being tested by attempting to design a

security kernel for a system with functional characteristics similar to Multics. The basic design is complete and some of the abstract programs have been written in the CLU language (which resulted from the completed task "High Level Description of System Functionality"). Work is under way to make the verification of the resulting system easier and on improving the efficiency of the resulting system without making the verification harder.

The hierarchical decomposition of the kernel into abstract levels is directly applicable to the certification of a kernel-based Multics. The CLU language has been suspended by SRI's SPECIAL language as the latter more readily lends itself to formal certification. This task has proved to be quite valuable to the Guardian programs.

Documentation of the methodology, design of the Multics-like system, and other results are in process.

7. Study of Multics Security Holes

A listing and report is made annually cataloging all known security flaws in the system, ways to violate the security of the system, or ways to crash the system. An attempt is made to analyze each flaw and to identify the general class of problem represented by the flaw.

One general class of security loop-hole was discovered and an automated search was made of the system to determine which modules might be susceptible to this particular attack. Several were found and repaired. After the repairs were made, the nature of the flaw was published to the development community so that the flaw would not be repeated.

The implementation technique developed for the Multics kernel will avoid the flaws uncovered by this task. The kernel must be proven correct and is therefore closely scrutinized during all stages of development.

8. Restructuring the Network Control Program

The Network Control Program (NCP) represents an ideal candidate to use in an experiment involving a multiple process implementation of a control algorithm, as discussed under "Multitasking in the User Ring". The Network Control Program is concerned with the flow of data to and from the ARPA Network. Its principle function is the management of a multiplexed communication path, which implies the management of multiplexed buffers. It was proposed that the flow of data between these various buffers be implemented using the fast processes in ring 0.

A related area is the possibility that common elements may be identified in the software that is required to handle different multiplexed communication streams. It is possible that there is a similar function in the software that interfaces the ARPANET and the 355. A buffer manager routine is an obvious example of a possible common routine. It is very interesting and profitable to identify these modules and to isolate them.

This could markedly reduce the amount of code within the kernel. Also, if all multiplexed communication streams could be handled by one set of kernel modules, such as the network server, this could result in great simplification of the kernel. This observation has lead to the start of a general re-examination of the way that I/O is done by the system. This research task has become the standard NCP in Multics until such time that a replacement version becomes available for the kernel-based Multics.

9. New I/O Buffer Strategy

This task involves the design and implementation of a new I/O buffering strategy which uses the virtual memory itself as a buffer. The task has become part of the task of restructuring the Network Control Program. The task arose from an attempt to increase the efficiency of transmitting data to and from the ARPA Network and at the same time to gain a more basic understanding of the interaction between Input/Output functions and virtual memory computer systems.

The result of this design is a buffer that uses the virtual memory and appears to be infinite in length. The use of the virtual memory eliminates any need to compact or otherwise manage the buffer area, thus reducing overhead. Since the buffer is in the process virtual address space, it is directly accessible to a user process. This avoids the copying of data to make it accessible.

It appears that this I/O buffer strategy can be exploited successfully for all devices for which the system nucleus is responsible. This unification of buffer management is a significant contribution to the certification project due to its reduction of bulk and complexity in the kernel. This task is still being formulated. No experimental data exists. When this task is further along in its development, a technical evaluation can be made.

10. Formulation of Criteria for Inclusion of Modules Within the Kernel

This task is an attempt to identify a set of general rules which will specify those modules which must be within the kernel

of an operating system. One approach is a study of the separation of policy from mechanism in a module.

As part of this task, a discussion of the criteria to be used for including portions of the page control system within the kernel has been produced. The intent is to perform a similar study for various other portions of the system and to attempt to evolve a general theory of the structure of a kernel from them.

Although this task shows merit as ` research activity, Honeywell does not have any current plans to utilize this effort for Project Guardian.

11. Study of System Error Recovery

The system's ability to recover from errors is related to the problem of system initialization since both must assure or certify that the system is in some known state. This task is interested in determining whether there are some particular structures for data bases and algorithms that make it much easier to assure that the data base is in fact consistent and correct.

A particular project under this task is the attempt to learn more about the structure of data bases through a comparative analysis of Multics and the Burroughs operating system.

This task has been combined with the study of the relationships between security and reliability below.

12. Study of Relationships between Security and Reliability

This task involves studying the relationship between the security of a system and the reliability of that system. In the Multics system, it is presumed that a system failure may have an unknown effect on the security status of the system; this is the reason that the system is shut down, salvaged, and restarted after every unexplained system failure. It is not obvious, however, that security is directly dependent on reliability. If it were possible to determine that certain classes of computer failure could not influence the security state of the system, then the two functions would have nothing to do with each other. More strongly, it is possible that a highly reliable system contains mechanisms that are not desirable from the viewpoint of security. For example, one way to increase the reliable storage of information on a system is to make several copies of that information; many copies, however, increase the probability that the information may be compromised.

A study of a variety of systems with high reliability goals was begun to improve understanding of the relationship of security and reliability. The project will investigate

mechanisms in the system in the light of their implications for the security of the information stored on that system. The task has been combined with the Study of System Error Recovery task. Honeywell is interested in this broad research tapes to the extent that the criteria for Guardian are satisfied. When the detailed results of this task are finalized, Honeywell will re-examine the total task. Until that time, Honeywell is unable to use this task for Project Guardian.

13. Removal of the Storage Hierarchy from Ring 0

The removal of the linker and of name space management from ring 0 can be considered the first two steps in restructuring the file system. The next logical step is the removal of directories from ring 0. However, this is not appropriate at this time. This study describes how directory control can be partitioned but remain within ring 0. The task will be renamed "Separating the Directory Hierarchy from the Segment Catalog" to reflect this redirection.

This research proposes that directory control be partitioned into two components, each with its own data base. There will be an unstructured segment catalog, indexed by unique ID. This catalog maintains the physical attributes for each segment such as file map, bit count, various date-time parameters, etc. There will also be a directory hierarchy which maintains the association between character string names and unique IDs. This hierarchy will also maintain the access control list for each segment. The principal change in the current proposal, relative to earlier proposals, is that the access control list will be stored in the directory hierarchy rather than in a separate access hierarchy.

This engineering approach for the restructure and removal of directory control from ring 0 is becoming an integral part of the evolving kernel design for Project Guardian. Honeywell is utilizing elements of this research task.

14. Support of User Defined Object Types

A directory can be considered as an object defined in terms of a lower level object, the segment. It is possible that the mechanisms that define the directory could be generalized to allow the definition of new object types defined by users. This sort of ability has been provided in systems that are based on capabilities, but not in systems that are based on access control lists. This task has been considering the question of whether user defined object types can be supported in a system such as Multics.

There are two projects in this area. First, there is the consideration of how extended objects can be supported using an appropriate combination of access control list and capability based mechanisms. Secondly, there is the study of the implementation of user defined objects in a pure access control list environment. This project is considering what policies for protection can be imposed upon these user defined extended objects. The research merit of this task is warranted. However, due to the extent of the required re-engineering, unknown performance impact, and its effect on security, this task is not presently required for Project Guardian.

15. Multics Performance Benchmark

As the tasks of simplifying the system are accomplished and further modifications to the system are proposed, it is important to be able to determine the performance effects. A stable and reliable performance meter is needed.

Two projects have been underway. One involves rework of the standard Multics performance benchmark. There is now a version of the benchmark that can be debugged on line and is extremely stable in the virtual cpu time required for the run. The other project involves creation of a system load generator. All test load generators which have existed for Multics in the past have suffered from one of two drawbacks, either they use absentee jobs to generate their load (which does not exercise the I/O system) or they require a large number of telephone data sets (which is very expensive). A load generator has been developed that can be run using the ARPANET to drive the test processes. This is useful as it includes a test of the I/O system.

Honeywell will use the well-tested standard-script performance benchmark for Project Guardian. As a result, this task to develop a low-generator, dynamic benchmark has become unnecessary for Project Guardian.

16. Independent Domains and Breakproof Services

This new task is to explore some of the implications of removing certain traditional supervisor functions from the Multics security kernel and to explore extending the function of the Multics protection mechanisms to allow multiple, independent domains to be part of one process.

A traditional supervisor includes many mechanisms that are not security sensitive simply to protect these mechanisms from accidental damage from user errors. To produce a security kernel for Multics, many such mechanisms are being moved out of the supervisor. By moving them to the user environment, mechanisms such as the linker, the reference name manager, and the search

rules become breakable, which could make the system harder to use. Fortunately, the Multics protection rings provide a place to protect non-kernel mechanisms that should be breakproof. They can execute in a ring, say ring 3, above the kernel but below the normal user ring. Because all data bases managed by these service mechanisms are private to a process, they are not part of the security kernel and need not be certified, yet they cannot be broken inadvertently by user errors. Part of this project is figuring out how to provide such breakproof services for Multics.

The second aspect of this project concerns protected subsystems. Multics has always supported user-defined protected subsystems, although the protection rings can provide only one way protection. It is not possible, however, to use the rings to protect both subsystems and breakproof services at the same time in the same process without making the breakproof services common to all subsystems in a process and therefore part of the security kernel for those subsystems. The essential difficulty is the total ordering of privilege implied by the protection rings. Thus, to provide breakproof services some other way must be found to protect subsystems, if the functionality of protected subsystems is to be maintained. The method being explored is simulating multiple independent domains (containing rings) in a process using multiple descriptor segments for a process. This project is just beginning.

The results of this task are too preliminary for a complete evaluation. However, Honeywell presently believes this effort may require extensive re-engineering to be useful for Guardian.

Section III

Summary

This paper has presented the status of the technical tasks in a research project to evolve the Multics supervisor into a security kernel capable of supporting current Multics functions. A sample of the specific tasks and investigations has been described. Findings from these tasks are being reviewed to determine applicability to design of the Multics kernel. In total, these research efforts contribute to the overall objective of determining the best way to reduce the size and complexity of the Multics software which will be certified as correct.

References

1. James P. Anderson, "Computer Security Technology Planning Study", ESD-TR-73-51, October 1972.

2. R. Schell, P. Downey, G. Popek, "Preliminary Notes on the Design of a Secure Military Computer System", MCI-73-1, January, 1972.

3. D. E. Bell, L. J. LaPadula, "Secure Computer Systems", ESD-TR-73-278, The MITRE Corporation, Bedford, Mass.

4. W. R. Price, "Implications of a Virtual Memory Mechanism for "Implementing Protection in a Family of Operating Systems", PhD Thesis, Carnegie-Mellon University, June, 1973.

5. E. L. Burke, "Synthesis of a Software Security System", MTP-154, The MITRE Corporation, Bedford, Mass.

6. D. E. Bell, E. L. Burke, "A Software Validation Technique for Certification: The Method", ESD-TR-75-54, The MITRE Corporation, Bedford, Mass, December, 1973.

7. L. Robinson, P. G. Neumann, K. N. Levitt, A. Saxena, "On Attaining Reliable Software for a Secure Operating System", 1975 International Conference on Reliable Software, Los Angeles, Ca, April, 1975.

8. W. L. Schiller, "Design of a Security Kernel for the PDP-11/45", The MITRE Corporation, Bedford, Mass, December, 1973.

9. W. L. Schiller, "The Design and Specification of a Security Kernel for the PDP-11/45", The MITRE Corporation, Bedford, Mass, March, 1975.

10. L. Smith, "Architectures for Secure Computing Systems", ESD-TR-75-51, The MITRE Corporation, Bedford, Mass, June, 1974.

11. J. C. Whitmore, A. Bensoussan, P. A. Green, A. M. Kobziar, J. A. Stern, "Design for Multics Security Enhancements", ESD-TR-74-176, Honeywell Information Systems, Inc., 1974.

12. Access Isolation Mechanism Pre-Release Documentation, Honeywell Information Systems, Inc., McLean, Virginia, August, 1975.

13. Michael D. Schroeder, "Security Kernel Evaluation for Multics (Interim Report)", ESD-TR-75-95, September 1975.

14. N. Adleman, J. Gilson et al, "Multics Security Integration Requirements, 1976 - 1980", to appear as ESD-TR-XXXX, March 1976.

15. W. L. Schiller, K. J. Biba, E. L. Burke, "The Top Level Specification of a Multics Security Kernel", Working Paper WP-20377, The MITRE Corporation, Bedford, Mass., August 1975.

16. J. A. Stern, "Multics Security Kernel Top-Level Specification", to appear as ESD-TR-XXXX, July 1976.

17. J. H. Saltzer and M. D. Schroeder, "The Protection of Information in Computer Systems," Proc IEEE 63, 9 (September 1975), pp 1278-1308.

18. P. A. Janson, "Removing the Dynamic Linker from the Security Kernel of a Computer Utility", MIT Project MAC Tech Rep. MAC-TR-132, June 1974.

19. P. A. Janson, "Dynamic Linking and Environment Initialization in a Multi-Domain Computation," ACM 5th Symp. on Operating Sys Principes, Austin, Texas, Nov. 1975.

20. R. G. Bratt, "Minimizing the Naming Facilities Requiring Protection in a Computer Utility", MIT Project MAC Tech Rep. MAC-TR-156, Sept. 1975.

21. B. S. Greenberg, "Experimental Analysis of Program Reference Patterns in the Multics Virtual Memory", MIT Project MAC Tech Rep MAC-TR-127, May 1974.

APPENDIX A

Air Force Electronic Systems Division Comments

Page 2, para 3, 2nd sentence: Isn't this true for Multics also?

Page 3, first line: A non-Honeywell reference would add credibility to this claim.

Page 10, paragraph 2: The first sentence is somewhat cryptic. Explain what is meant by "to lie convincingly".

Page 10, para 4, 2nd sentence: The reason for the necessity indicated here is unclear.

Page 11, para 2: The intent of the last sentence is unclear.

Page 11, para 4: Rephrase the fifth sentence or else make two sentences out of it. The meaning is not clear. Is there only one such language?

Page 12, para 1, line 2: "Easier" than what other mode? To what is the comparison being made? Same for "more readily".

Page 12, para 2: Honeywell's "review of this task is unclear. If the scheme requires an I/O controller with more capabilities than in the current Multics, and will not be implemented in the near future, why will results of the work be incorporated into the kernel design? Which results?

Page 13, para 3, 2nd sentence: It is unclear how the conclusion was derived.

Page 14, para 2, 3rd sentence: Will the low level multiplexor need to make use of the system's virtual memory facilities if there are more virtual processors than real processors? If not, then what does fixing the number of virtual processors in advance buy you over not fixing the number in advance? What is the difference in operation between the high level and low level multiplexors?

Page 14, para 2, 4th sentence: It is unclear what "multiplexes" are. Should the word be "multiplexors"?

Page 15, para 2: Is certified software not needed because the ACL software is certified, or is certified software simply not needed?

Page 16, para 2, sentence 3: The meaning of the "xxx" in this sentence is unclear.

Page 16, para 2: Typographical error: "This work will be further reviewed a additional data becomes available and then overall kernel design progresses".

Page 17, para 1: Honeywell's review of task 5 here is no complete and is unclear.

Page 17, para 3, line 4: Please explain what is meant by "the CLU language has been suspended".

Page 17, Section 7: Briefly describe that nature of the flaw.

Page 17, para 7: More should be said about this task. Is it true that all holes found would be corrected by a verified kernel? Were there no covert channels discovered?

Page 18, para 2-4: What has task8 to do with a kernel for a secure Multics?

Page 18, para 4, last sentence: A research task could not become the standard NCP since the NCP is a software system and a research task is not.

Page 19, para 7: Why the Burroughs operating system? More rationale is required.

Page 20, para 2, line 8: It is unclear what "... broad research tapes ..." are.

Page 21, para 5, 1st sentence: "... determine the performance effects" of what and/or on what?

Page 22, para 2, line 3: Explain what a "low-generator, dynamic benchmark" is.